TITLE OF THE INVENTION

DECODING APPARATUS, CONTROL METHOD THEREFOR, AND STORAGE MEDIUM

FIELD OF THE INVENTION

The present invention relates to an apparatus of decoding a moving picture, a control method therefor, and a storage medium.

BACKGROUND OF THE INVENTION

As a color still picture encoding method, JPEG, an international standard, has widely been known. As a compression method having a larger number of functions than JPEG, a new encoding method based on wavelet transform, i.e., so-called JPEG 2000 is being standardized.

Further, attempts are made to apply the wavelet transform to moving picture encoding.

As the processing speed of recent CPUs increases, processing of a moving picture compressed using the wavelet transform technique can be realized. In a moving picture, data are successive so decoding processing must be completed within a predetermined time.

SUMMARY OF THE INVENTION

The present invention has been made to overcome the conventional drawbacks, and has as its object to eliminate visible flaws between frames even if several frames cannot be decoded.

To achieve the above object, for example, a decoding apparatus of the present invention has the following arrangement.

According to the present invention, the foregoing object is attained by providing a decoding apparatus for decoding an input signal which includes a code stream and data having time management information about decoding of the code stream, comprising: buffering means for buffering the input signal; decoding means for reading out the input signal from the buffering means, decoding the input signal, and writing decoded data in a predetermined memory; output means for outputting the decoded data from the predetermined memory; and control means for controlling the buffering means and the decoding means on the basis of the time management information.

According to another aspect of the present invention, the foregoing object is attained by providing a method of controlling a decoding apparatus for decoding an input signal which includes a code stream and data having time management information about decoding of the code stream, comprising: a buffering step of buffering the input signal in a predetermined buffer; a decoding step of reading out the input signal from the predetermined buffer, decoding the input signal, and writing decoded data in a predetermined memory; a output step of outputting the decoded data from the predetermined memory; and the control step of controlling at least one of the buffering step, the decoding step, and the output step on the basis of the time management information.

In still another aspect of the present invention, the foregoing object is attained by providing a storage medium which stores program codes functioning as a decoding apparatus for decoding an input signal which includes a code stream and data having time management information about decoding of the code stream, comprising: a program code of the buffering step of buffering the input signal in a predetermined buffer; a program code of the decoding step of reading out the input signal from the predetermined buffer, decoding the input signal, and writing decoded data in a predetermined memory; a program code of the output step of outputting the decoded data from the predetermined memory; and a program code of the control step of controlling at least one of the buffering step, the decoding step, and the output step on the basis of the time management information.

Other features and advantages of the present invention will be apparent from the following description taken in conjunction with the accompanying drawings, in which like reference characters designate the same or similar parts throughout the figures thereof.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and constitute a part of the specification, illustrate embodiments of the invention and, together with the description, serve to explain the principles of the invention.

Fig. 1 is a block diagram showing the schematic arrangement of a decoding apparatus in the first embodiment of the present invention;

Fig. 2 is a flow chart showing decoding processing of the decoding apparatus in the first embodiment of the present invention;

Fig. 3A is a view showing decoding processing in 2x-speed playback according to the first embodiment of the present invention;

Fig. 3B is a view showing an example of an image displayed in normal playback according to the first embodiment of the present invention;

Fig. 3C is a view showing an example of an image displayed in 2x-speed playback according to the first embodiment of the present invention;

Fig. 4A is a view showing an entire encoded stream which should be originally decoded in the first embodiment of the present invention;

Fig. 4B is a view showing data decoded partway owing to a short processing time in the first embodiment of the present invention;

Fig. 5 is a flow chart showing decoding processing of a decoding apparatus in the second embodiment of the present invention;

Fig. 6A is a view showing decoding processing in 4x-speed playback according to the second embodiment of the present invention;

Fig. 6B is a view showing an example of an image displayed in 4x-speed playback according to the second embodiment of the present invention;

Fig. 7 is a block diagram showing the arrangement of a conventional encoder;

Fig. 8A is a block diagram showing the arrangement of a discrete wavelet transform unit 702;

Fig. 8B is a block diagram showing the arrangement of a processing unit 802 in Fig. 8A;

Fig. 8C is a view showing the layout of 2-level transform coefficient groups obtained by two-dimensional transform processing;

Fig. 9 is a view for explaining the operation of an entropy encoding unit 704;

Fig. 10A is a view showing the structure of an entire code stream;

Fig. 10B is a view showing a structure of a main header MH;

Fig. 10C is a view showing a structure of a tile header TH;

Fig. 10D is a view showing a structure of a bit stream;

Fig. 11A is a view showing the entire structure of a code stream different from that of Fig. 10A;

Fig. 11B is a view showing a structure of the main header MH;

Fig. 11C is a view showing a structure of the tile header TH;

Fig. 11D is a view showing a structure of the bit stream;

Fig. 12 is a block diagram showing the arrangement of a conventional decoder;

Fig. 13 is a view for explaining the operation of an entropy decoding unit 1202;

Fig. 14A is a block diagram showing the arrangement of an inverse discrete wavelet transform unit 1204;

Fig. 14B is a block diagram showing the arrangement of a processing unit 1402 in Fig. 14A;

Fig. 15A is a view showing a structure of a code stream;

Fig. 15B is a view showing the relationship between each subband and the size of a displayed image;

Fig. 16A is a view showing a structure of a code stream different from that of Fig. 15A;

Fig. 16B is a view showing an example of a displayed image;

Fig. 17 is a block diagram showing a conventional decoding apparatus of decoding multiplexed video and audio data;

Fig. 18A is a view for explaining conventional processing from reception to display of a packet;

Fig. 18B is a view for explaining conventional processing from reception to display of a packet in 2x-speed playback;

Fig. 19A is a view for explaining decoding processing in the second embodiment of the present invention;

Fig. 19B is a view showing an example of an image displayed according to the second embodiment of the present invention;

Fig. 20 is a flow chart showing decoding processing of a decoding processing in the third embodiment of the present invention;

Fig. 21A is a view showing an example of dividing one frame into nine tiles in the third embodiment of the present invention;

Figs. 21B to 21J are views showing examples of shuffling tiles according to the third embodiment of the present invention;

Fig. 22 is a graph for explaining the

relationship between the image quality and the data amount;

Fig. 23 is a flow chart showing decoding processing of a decoding apparatus in the fourth embodiment of the present invention;

Fig. 24A is a view for explaining decoding processing by 2x-speed playback in the fourth embodiment of the present invention;

Fig. 24B is a view showing an example of an image displayed in normal playback according to the fourth embodiment of the present invention;

Figs. 24C and 24D are views showing examples of an image displayed in 2x-speed playback according to the fourth embodiment of the present invention;

Fig. 25A is a view showing an entire encoded stream which should be originally decoded in the fourth embodiment of the present invention;

Fig. 25B is a view showing data decoded partway owing to a short processing time in the fourth embodiment of the present invention;

Fig. 26 is a flow chart showing decoding processing of a decoding apparatus in the fifth embodiment of the present invention;

Fig. 27A is a view for explaining decoding processing by 4x-speed playback in the fourth embodiment of the present invention;

Fig. 27B is a view showing an example of an image displayed in 4x-speed playback according to the fourth embodiment of the present invention;

Fig. 28A is a view for explaining decoding processing by 4x-speed playback in the fifth embodiment of the present invention;

Fig. 28B is a view showing an example of an image displayed in 4x-speed playback according to the fifth embodiment of the present invention;

Fig. 29 is a flow chart showing decoding processing of a decoding apparatus in the sixth embodiment of the present invention;

Fig. 30 is a graph for explaining the relationship between the image size and the data amount;

Fig. 31 is a view for explaining the relationship between the frame rate and the data amount;

Fig. 32 is a table showing an example of a table representing the relationship between the playback speed of high-speed playback and the lowest image quality in the second embodiment of the present invention;

Fig. 33 is a table showing an example of a table representing the relationship between the playback speed of high-speed playback and the smallest image size in the fifth embodiment of the present invention; and

Fig. 34 is a block diagram showing an example of a personal computer functioning as a decoding apparatus of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Preferred embodiments of the present invention will be described in detail below with reference to the accompanying drawings. Program codes complying with a flow chart described in each embodiment are stored in a memory formed from a RAM or ROM in a decoding apparatus according to the embodiment, and read out and executed by a CPU.

[First Embodiment]

A conventional encoding method based on the wavelet transform will be explained.

(Encoder)

In Fig. 7 which is a block diagram showing a conventional encoder, reference numeral 701 denotes an image input unit; 702, a discrete wavelet transform (DWT) unit; 703, a quantization unit; 704, an entropy encoding unit; and 705, a code output unit.

Pixel signals forming an image to be encoded are input to the image input unit 701 in a raster scan order, and an output from the image input unit 701 is input to the discrete wavelet transform unit 702. In the following description, an image signal expresses a monochrome multilevel image. To encode a plurality of color components, like a color image, R, G, and B color components or luminance and chromaticity components are compressed as a single color component.

The discrete wavelet transform unit 702 performs two-dimensional discrete wavelet transform processing for an input image signal, and calculates and outputs a transform coefficient. Fig. 8A shows the basic arrangement of the discrete wavelet transform unit 702 in which input image signals are stored in a memory 801, sequentially read out and subjected to transform processing by a processing unit 802, and rewritten in the memory 801. The processing unit 802 has a processing arrangement shown in Fig. 8B. In Fig. 8B, the input image signal is separated into even- and odd-numbered address signals by a combination of a delay element and down samplers, and subjected to filter processing by two filters p and u. Reference symbols s and d in Fig. 8B denote low- and high-pass coefficients in 1-level decomposition for one-dimensional image signals, and are calculated by

$$d(n) = x(2*n+1) - \mathrm{floor}((x(2*n)+x(2*n+2))/2) \quad (1)$$

$$s(n) = x(2*n) + \mathrm{floor}((d(2-1)+d(n))/4) \quad (2)$$

where x(n) is the image signal to be transformed, and floor(r) is a maximum integer value smaller than r.

By the above processing, image signals undergo one-dimensional discrete wavelet transform processing. The two-dimensional discrete wavelet transform sequentially executes a one-dimensional transform in the horizontal and vertical directions of an image. Details of the two-dimensional discrete wavelet transform are well known, and a description thereof will be omitted.

Fig. 8C shows the layout of 2-level transform coefficient groups obtained by two-dimensional transform processing. An image signal is decomposed into coefficient streams HH1, HL1, LH1,..., LL having different frequency bands. In the following description, these coefficient streams will be called subbands. The coefficient of each subband is output to the subsequent quantization unit 703.

The quantization unit 703 quantizes the input coefficient at a predetermined quantization step, and outputs an index (quantization index) for the quantized value. The quantization unit 703 executes quantization by

$$q = \mathrm{sign}(c)\mathrm{floor}(\mathrm{abs}(c)/\Delta) \quad (3)$$

$$\mathrm{sign}(c) = 1; c >= 0$$

(4)

$$\mathrm{sign}(c) = -1; c <= 0 \quad (5)$$

where c is a coefficient to be quantized, and floor(X) is a maximum integer value smaller than X. The Δ value can be 1. In this case, no quantization is actually done, and a transform coefficient input to the quantization unit 703 is directly output to the subsequent entropy encoding unit 704.

The entropy encoding unit 704 decomposes the input quantization index into bit planes, performs binary arithmetic encoding in units of bit planes, and outputs a code stream.

Fig. 9 is a view for explaining the operation of the entropy encoding unit 704. In this example, three nonzero quantization indices exist in the regions of a 4 x 4 subband, and have values of +13, -6, and +3, respectively. The entropy encoding unit 704 scans the respective subband regions to obtain a maximum value M among all the quantization indices, and calculates the number S of bits necessary for expressing the maximum quantization index by

$$S = \mathrm{ceil}(\log_2(\mathrm{abs}(M)) \qquad (6)$$

where ceil(x) is a minimum integer value among integers equal to or larger than x. In Fig. 9, the maximum quantization index value M is 13. From equation (6), S is 4, and 16 quantization indices in a sequence are processed in units of four bit planes, as shown in Fig. 9.

The entropy encoding unit 704 executes binary arithmetic encoding for the respective bits of the most significant bit plane (represented by MSB in Fig. 9), and outputs them as a bit stream. The entropy encoding unit 704 switches the bit plane to one lower by one level, and then encodes the respective bits of the bit plane and outputs them to the code output unit 705 until the target bit plane reaches the least significant bit plane (represented by LSB in Fig. 9). At this time, the code of each quantization index is entropy-encoded immediately after a nonzero bit is first detected in bit plane scan.

Fig. 10A is a schematic view showing the structure of a code stream generated and output in this manner. In Fig. 10A, MH represents a main header; TH, a tile header; and BS, a bit stream. As shown in Fig. 10B, the main header MH is constituted by pieces of information including the size of an image to be encoded (the numbers of pixels in the horizontal and vertical directions), the size of a tile obtained by dividing the image into a plurality of rectangular regions, the number of components representing the number of color components, and component information representing the size of each component and the bit precision. When the image is not divided into tiles, the tile and image sizes take the same values. When the target image is a monochrome multilevel image, the number of components is 1.

Fig. 10C shows the structure of the tile header TH. The tile header TH is constituted by pieces of information representing a tile length including the bit stream length and header length of a tile, and an encoding parameter for the tile. The encoding parameter includes the level of discrete wavelet transform and the type of filter.

Fig. 10D shows the structure of the bit stream. In Fig. 10D, bit streams are gathered in units of subbands and arranged in an ascending order of resolution from the bit stream of a subband whose resolution is the lowest. Bit planes are arranged from an upper bit plane to a lower bit plane in each subband, and codes are arranged in each bit plane.

Fig. 11A shows the entire structure of a code stream different from that of Fig. 10A. MH represents a main header; TH, a tile header; and BS, a bit stream. As shown in Fig. 11B, the main header MH is constituted by pieces of information including the size of an image to be encoded (the numbers of pixels in the horizontal and vertical directions), the size of a tile obtained by dividing the image into a plurality of rectangular regions, the number of components representing the number of color components, and component information representing the size of each component and the bit precision. When the image is not divided into tiles, the tile and image sizes take the same values. When the target image is a monochrome multilevel image, the number of components is 1.

Fig. 11C shows the structure of the tile header TH. The tile header TH is constituted by pieces of information representing a tile length including the bit stream length and header length of a tile, and an encoding parameter for the tile. The encoding parameter includes the level of discrete wavelet transform and the type of filter. Fig. 11D shows the structure of the bit stream. In Fig. 11D, bit streams are gathered in units of bit planes and arranged from an upper bit plane to a lower bit plane. In each bit plane, the encoding results of the bit planes of quantization indices in each subband are sequentially arranged in units of subbands. In Fig. 11D, S represents the number of bits necessary for expressing the maximum quantization index. A code stream generated in this manner is output to the code output unit 705.

The compression ratio of an entire image to be encoded can be controlled by changing the quantization step Δ. As another method, the entropy encoding unit 704 can also control (discard) the lower bit of a bit plane to be encoded in accordance with a necessary compression ratio. In this case, all the bit planes are not encoded, bit planes corresponding in number to a desired compression ratio are encoded from an upper bit plane, and the encoded bit planes are included in the final code stream.

(Decoder)

A method of decoding a code stream attained by the above-described encoder will be explained.

Fig. 12 is a block diagram showing the

arrangement of the decoder. Reference numeral 1201 denotes a code input unit; 1202, an entropy decoding unit; 1203, an inverse quantization unit; 1204, an inverse discrete wavelet transform unit; and 1205, an image output unit.

The code input unit 1201 receives the above-described code stream, analyzes the header included in it, extracts a parameter necessary for subsequent processing, and if necessary, controls the processing flow or outputs the parameter to the subsequent processing unit. The bit stream included in the code stream is output to the entropy decoding unit 1202.

The entropy decoding unit 1202 decodes the bit stream in units of bit planes, and outputs the decoded stream. This decoding procedure is shown in Fig. 13. Fig. 13 shows the flow of sequentially decoding one region of a subband to be decoded in units of bit planes and finally decoding the quantization index. Bit planes are decoded in the direction indicated by an arrow in Fig. 13. The decoded quantization index is output to the inverse quantization unit 1203.

The inverse quantization unit 1203 reconstructs the discrete wavelet transform coefficient from the input quantization index by:

$$c' = \Delta * q \; ; \; q \neq 0 \qquad (7)$$

$$c' = 0 \quad ; \; q = 0 \qquad (8)$$

where q is the quantization index, Δ is the quantization step and has the same value as that used in encoding, and c' is the decoded transform coefficient obtained by decoding the coefficient s or d in encoding. The transform coefficient c' is output to the subsequent inverse discrete wavelet transform unit 1204.

Fig. 14A is a block diagram showing the arrangement of the inverse discrete wavelet transform unit 1204. In Fig. 14A, the input transform coefficient is stored in a memory 1401. A processing unit 1402 executes the two-dimensional inverse discrete wavelet transform by performing a one-dimensional inverse discrete wavelet transform, sequentially reading out transform coefficients from the memory 1401, and processing them. The two-dimensional inverse discrete wavelet transform is executed by procedures reverse to those of the forward transform. Details of the two-dimensional inverse discrete wavelet transform are well known, and a description thereof will be omitted. Fig. 14B shows the processing block of the processing unit 1402. Input transform coefficients are subjected to two filter processes by u and p, up-sampled, and superimposed on each other, thereby outputting an image signal x'. These processes are performed by

$$x'(2*n) = s'(n) - \text{floor}((d'(n-1)+d'(n))/4)$$

(9)

x'(2*n+1) = d'(n) + floor((x'(2*n)+x'(2*n+2))/2)

(10)

In this case, the forward and backward discrete wavelet transform processes by equations (1), (2), (9), and (10) satisfy perfect reconstruction conditions. Thus, the decoded image signal x' coincides with the signal x of the original image as far as the quantization step Δ is 1 and all the bit planes are decoded in bit plane decoding.

By this processing, the image is restored and output to the image output unit 1205. The image output unit 1205 may be an image display device such as a monitor or a storage device such as a magnetic disk.

The image display form in restoring and displaying an image by the above-mentioned procedures will be described with reference to Figs. 15A and 15B. Fig. 15A shows an example of a code stream whose basic structure is based on Figs. 10A and 10D. When an entire image is set as one tile, the code stream includes only one tile header and one bit stream. As shown in Fig. 15A, codes are arranged in a bit stream BS0 in an ascending order of resolution from LL which is a subband corresponding to the lowest resolution.

The decoder sequentially reads bit streams, and when decoding codes corresponding to each subband, displays the image. Fig. 15B shows the correspondence between each subband and the size of a displayed image. In this example, the two-dimensional discrete wavelet transform is done to two levels. When only LL is decoded and displayed, an image made up of pixels reduced in number to 1/4 the original image in the horizontal and vertical directions is restored. When bit streams are further read and all the subbands of level 2 are restored and displayed, an image made up of pixels reduced in number to 1/2 the original image in the respective directions is restored. When all the subbands of level 1 are restored and displayed, an image having the same number of pixels as that of the original image is restored.

Another image display form in restoring and displaying an image by the above-mentioned procedures will be described with reference to Figs. 16A and 16B. Fig. 16A shows an example of a code stream whose basic structure is based on Figs. 11A and 11D. When an entire image is set as one tile, the code stream includes only one tile header and one bit stream. As shown in Fig. 16A, codes are arranged from the most significant bit plane to the least significant bit plane.

The decoder sequentially reads bit streams, and when decoding the codes of each bit plane, displays the image. Fig. 16B shows an example of a change in the quality of a displayed image when decoding is sequentially done from an upper bit plane. When only the upper bit plane is decoded, only the feature of the entire image is displayed. As lower bit planes are decoded, the image quality is improved stepwise. When the quantization step Δ is 1 in quantization, the image displayed after all the bit planes are decoded is completely the same as the original image.

In this prior art, lower bit planes to be decoded by the entropy decoding unit 1202 can be limited (ignored) to decrease the encoded data amount to be received or processed, thereby controlling the compression ratio. This enables obtaining a decoded image having a desired image quality from only a necessary amount of encoded data. The prior art can also realize reversible encoding/decoding in which a restored image coincides with its original image when the quantization step Δ is 1 in quantization and all the bit planes are decoded in decoding.

Fig. 17 shows a conventional decoding apparatus of decoding multiplexed video and audio data. A DEMUX unit 1701 demultiplexes video and audio streams. The data unit output to each of encoding buffers 1702a and 1702b is a sync processing unit generally called an access unit. Letting a frame be a processing unit, the processing proceeds such that the encoding buffers 1702a and 1702b receive data in units of frames, decoding units 1703a and 1703b decode the data, the decoded data are written in memories 1704a and 1704b, and output units 1705a and 1705b display the data. The access unit includes, at the header, time management information, which is used for sync management.

Fig. 18A shows timings from reception to display of a packet. The first packet received at timing t = t00 takes a time D1 for decoding processing and is displayed for a time C1. Similarly, the second packet received at timing t = t40 takes a time D2 for decoding processing and is displayed for a time C2. If decoding processing ends within a predetermined time, the same data as that on the encoding side is decoded with a predetermined delay time. For descriptive convenience, the buffering size is one frame. Processing must end till the read timing of the next frame data, which is the decoding limit time. In this case, the limit time described at the header is (t40 - t00), and the condition of displaying the first frame is D1 ≦ (t40-t00). In general, the processing time is designed to be much shorter than the limit time. However, when decoding is done by software installed in a general-purpose computer, 100% of the CPU performance cannot always be assigned to this processing due to execution of another application. To read data once stored in a hard disk at a high speed, 2x-speed playback requires two or more times the processing speed, and 3x-speed playback requires three or more times the processing speed. At this time, time management information within a packet header must be rewritten in accordance with the playback speed, and then processing is executed.

A problem when the processing speed is insufficient will be explained with reference to Fig. 18B by exemplifying 2x-speed playback. The first packet received at timing t = t00 takes the time D1 for decoding processing and is displayed for the time C1. At this time, the processing limit time for the first packet is (t20-t00) or (t40-t00)/2. Since D1 ≦ (t40-t00), the first packet can be displayed. However, as for the second packet received at timing t = t20, the time D2 for decoding processing is D2 > (t40-t20), so decoding processing of the next packet must start before decoding processing of the second packet ends. In this case, data which is being decoded is not written in the memory and holds its state until the next write is done. That is, the first data is kept displayed, and the display time is the time C2 during which the second data should be displayed. The third packet received at timing t = t40 takes the time D3 for decoding processing. The result can be displayed because D3 ≦ (t60-t40), and the third packet is displayed for a time C3. As for the fourth packet, the result can be displayed because D4 ≦ (t80-t40), and the fourth packet is displayed for a time C4.

In conventional decoding processing, if decoding cannot be completed owing to high-speed search, discontinuity is generated between frames, and visual flaws occur in the time axis direction. A decoding apparatus of the first embodiment for solving this problem will be described below.

Fig. 1 is a block diagram showing the schematic arrangement of the decoding apparatus in the first embodiment. Reference numeral 101 denotes a DEMUX unit; 102a and 102b, decoding buffers; 103a and 103b, decoding units; 104a and 104b, memories; 105a and 105b, output units; and 106a and 106b, sync controllers. The units 102a to 106a (to be referred to as decoder 1) and the units 102b to 106b (to be referred to as decoder 2) have the same arrangement, which means that a plurality of decoding processes are performed parallel. Generally, video decoding and audio decoding correspond to these decoding processes. One of the features of the first embodiment is that decoders 1 and 2 adopt the sync controllers. Decoder 1 will be explained mainly for the sync controller. Note that it is easily understood that the following description also applies to decoder 2.

The sync controller 106a detects the times at which the header of an access unit should be read and decoding 1 should end processing. If decoder 1 cannot complete processing within a necessary processing time, the following control is done for the decoding buffer 102a and decoding unit 103a. The timing of control and its processing contents will be described with reference to Fig. 2 showing the flow chart of this processing. In this case, the access unit is a frame.

The first data is read in step S201, and decoding of the frame data starts in step S202. This processing means a state wherein the decoding unit 103a reads out data from the decoding buffer 102a on demand and decodes them in Fig. 1. Until decoding processing ends at the branch in step S203, whether the current time exceeds the processing limit time assigned to this packet is monitored in step S204. For convenience, the time till the reception timing of the next frame may be assumed as an assigned maximum decoding processing time. Within the assigned processing limit time, the current data is kept decoded. If a bit stream is SNR scalable, respective subband data are arranged from the low-frequency side to the high-frequency side for each bit plane. An example is bit planes included in the code stream shown in Fig. 16A. If the number of bits assigned to each subband changes, the number of subbands on the MSB side is smaller than that on the LSB side. In this example, the number of subbands on the MSB side is three, LL, HL2, and LH2. The first current data in Fig. 2 is an LL band of a bit plane (S-1).

After step S205, data of the next subband, i.e., HL2 is set as new data to be processed in step S206. The flow passes through this loop again to process LH2, and decoding of all the data included in the bit plane (S-1) ends. After step S207, the flow shifts to processing of a bit plane BitsS-2 in step S208. The flow repeats this loop to end decoding up to a bit plane 0 and end decoding of all the data of this frame.

In step S209, the current frame data is written in the memory. The display timing complies with another information in the packet header. At the reception timing of the next frame data in step S210, whether the next frame data exists is checked in step S211. If YES in step S211, the next frame data is set as current frame data in step S212, and the flow returns to processing of step S202; if NO in step S211, a series of processes end. If it is determined in step S204 that processing did not end within the limit time, only decoded data are written in the memory in step S213. The display timing complies with time management information in the packet header. Since decoding processing ends within the limit time, the display time defined in the header coincides with the actual data display time. This relationship will be explained with reference to Figs. 3A to 3C.

Fig. 3A shows 2x-speed playback, similar to Fig. 18B. The first packet received at timing t = t00 takes a time D1 for decoding processing and is displayed for a time C1. The second packet received at timing t = t20 takes a time D2 for decoding processing, but all the data cannot be decoded within this time because D2 > (t40-t20). Fig. 4A shows an entire encoded stream which should be originally decoded, and Fig. 4B shows data decoded partway owing to a short processing time. In this example, the final bit plane Bit0, and HL1, LH1 and HH1 of the bit plane Bit1 are not decoded. Hence, the image of this frame is slightly lower in image quality than other frames. Fig. 3C is a view for explaining the image quality in 2x-speed playback. The image quality of the second frame is slightly lower than the image quality in normal playback in Fig. 3B. However, the number of frames to be played back does not decrease, so that a smooth motion can be reproduced similarly to normal playback in Fig. 3B. In this manner, if the processing time is short, only decoded data are written in the memory to guarantee the image quality of at least the decoded data.

Fig. 6A shows 4x-speed playback. As for the first packet received at timing t = t00, all the data cannot be decoded within the time D1, and only data which can be decoded till the next packet reception timing t = t10 are displayed for the time C1. This also applies to D2, D3, and D4. In this example, as all the four packets cannot be decoded within their processing times, the images of the frames are displayed at a low image quality. However, the number of frames to be played back does not decrease even at a 4x speed, so that a smooth motion can be reproduced.

As described above, image continuity can be maintained by the above processing for each packet though the image quality slightly changes between frames.

[Second Embodiment]

In the second embodiment, even a frame whose processing time cannot fall within a predetermined time maintains image quality at a predetermined or higher level. A decoding apparatus in the second embodiment has the same arrangement as that in Fig. 1.

The flow chart of decoding processing of the decoding apparatus in the second embodiment will be described with reference to Fig. 5. Note that a description of the same processing as in the flow chart shown in Fig. 2 will be omitted. Assuming that processing of step S201 has already ended in processing of step S301, processing of step S201 is not shown in Fig. 5.

If the end of decoding of the current data is determined in step S302, the same processing as processing from steps S205 to S211 is executed. Whether the processing time has reached the limit time is checked in step S303. If YES in step S303, whether the image quality of the decoded data is higher than the lowest image quality set in advance is determined in step S304. The image quality is set in advance such that all data must be decoded up to the bit plane (S-1) or data must be decoded up to the most significant bit plane of all the subbands. The lowest image quality is preferably set in accordance with the processing performance of the decoding unit, which will be described later. If YES in step S304, the flow advances to step S305, and the decoding result is written in a memory 104a.

Similar to Fig. 2, the flow waits for the reception timing of the next frame data in step S306. If the next frame data exists in step S307, the next frame data is set as current frame data in step S308, and decoding processing continues. If NO in step S307, the current frame data is determined as a final frame, and a series of processes end.

If NO in step S304, the flow waits for the reception timing of the next frame data in step S309. If the next frame data exists in step S310, the limit time is changed in step S311. The limit time is changed by newly adding time management information within the packet header. That is, the current frame is processed within the total time of the processing time of the current frame and the processing time of the next frame. In step S312, the next frame data is discarded, and data in a decoding buffer 102a is held.

This relationship will be explained by exemplifying 4x-speed playback in Fig. 19A. Fig. 19B shows a display example of an image in this case. In decoding processing of the first frame in Fig. 19A, all encoded data cannot be decoded, but the image quality is higher than the lowest image quality, and the first frame image is displayed for the time C1. In decoding processing of the second frame, data cannot be decoded even with the lowest image quality within the limit time D2, so that the decoding time of the second frame is assigned up to the decoding limit time t of the third frame = t30. The display time C2 of the second frame image corresponds to two frames. Hence, the third frame image is not displayed, but the image quality of the second frame image is held.

When the lowest image quality is not guaranteed, the frame frequency (the number of frames displayed per second), i.e., the display frame rate increases as the speed of high-speed playback increases. In some cases, the image quality is preferably improved even by decimating or down-sampling frames in terms of the processing performance of the decoding unit, the display performance of the display, and visual effects. Fig. 31 is a graph showing an example of the relationship between the data amount per frame and the display frame rate. To increase the display frame rate, the data amount per frame must be decreased. To increase the data amount per frame, the display frame rate must be decreased. The data amount and image quality have a relationship as shown in Fig. 22. In this case, decreasing the data amount per frame degrades the image quality. The lowest image quality is set by balancing the smoothness of the frame motion and the image quality of each frame. As a simple implementation means, the relationship between the playback speed of high-speed playback and the lowest image quality is held as a table. Fig. 32 shows an example of the table. This example assumes data of 8-bit precision. All the bits are guaranteed up to a 2x speed, upper six bits are guaranteed up to a 4x speed, and upper four bits are guaranteed up to a 6x speed. For a speed higher than the 6x speed, an image quality of upper three bits is guaranteed.

As described above, the decoding apparatus of the second embodiment always manages the image quality of a frame to be decoded and always generates an image with a predetermined or higher image quality.

[Third Embodiment]

A decoding apparatus in the third embodiment has the same arrangement as that in Fig. 1, but has different processing control contents. Details of the processing will be described with reference to the flow chart of decoding processing in the third embodiment shown in Fig. 20. A description of the same processing as that in the flow chart shown in Fig. 2 will be omitted. In the case, the processing unit is not a frame but a tile. The tile is obtained by breaking up a frame into a plurality of regions, and processing for each region is repeated by the number of tiles.

The first plane of the first tile in the current frame is processed for each band (loop of steps S2003 → S2004). After processing of all the bands ends, processing is repeated for the next plane (loop of steps S2005 → S2006). After processing of all the planes ends, processing is repeated for the next tile (loop of steps S2007 → S2008). This processing is repeated for all the tiles in the frame.

If the lowest image quality in high-speed playback is set in processing in units of tiles, a region where the image is updated may be localized owing to the difference in data amount between tiles on the screen. This is prevented by setting the next tile data at random in step S2008, which will be explained with reference to Figs. 21A to 21J.

Fig. 21A shows an example of dividing one frame into nine tiles. Figs. 21B to 21J show shuffling examples of nine patterns using these tiles. The decoding priority is T1 → T2 → ... → T9. In shuffling, the priority is sequentially changed at the same position of each tile on the screen. When decoding processing cannot be completed and data T9 fails to be decoded, data below the center in Fig. 21B, upper right data in Fig. 21C, and data left to the center in Fig. 21D cannot be decoded. Accordingly, successive failure in decoding data at the same position can be prevented.

When only a tile T1 can be processed by the decoder within a given time, the display is updated at a position above the center in Fig. 21B, a position below the center in Fig. 21C, and an upper right position in Fig. 21D. In this way, data is updated at different positions for respective frames, and data goes the round by nine frames. Shuffling is implemented by reading the header of each packet and setting the decoding start position of the decoding unit in the decoding buffer. At this time, the read position of bit stream data and the write position of decoded data in the memory are made to coincide with each other.

As described above, decoding processing of the decoding apparatus according to the third embodiment can prevent the image at the same position from failing in decoding successively because the tile position not decoded changes in playing back a decoded frame.

[Fourth Embodiment]

A decoding apparatus in the fourth embodiment has the same arrangement as that in Fig. 1, but has different processing control contents. The control timing and processing contents are shown in the flow chart of Fig. 23. In this embodiment, the access unit is a frame.

The first data is read in step S2301, and decoding of the frame data starts in step S2302. This processing means that a decoding unit 103a or 103b in Fig. 1 reads out data from a decoding buffer 102a or 102b on demand and decodes the data.

Until decoding processing ends at the branch in step S2303, whether the current time exceeds the processing limit time assigned to this packet is monitored in step S2304. For convenience, the time till the reception timing of the next frame may be assumed as an assigned maximum decoding processing time. Within the assigned processing limit time, the current data is kept decoded. If a bit stream is space-scalable, respective bit plane data are arranged from the upper-bit side to the lower-bit side for each subband. An example is subbands included in the code stream shown in Fig. 15A.

If the end of decoding of the current data is determined in step S2303, the flow advances to step S2305, and whether all the planes have been processed is checked. After step S2305, the next plane data, i.e., BitS-2 is set as new data to be processed in step

S2306. After this loop is done for all the planes, processing for the first subband LL ends. After step S2307 of determining whether all the bands have been processed, processing shifts to the next subband HL2 in step S2308. After the above loop is repeated and decoding proceeds up to the final subband HH1, all the data of the frame have been decoded.

In step S2309, the current frame data is written in the memory. The display timing complies with another information in the packet header. At the reception timing of the next frame data in step S2310, the flow advances to step S2311, and whether the next frame data exists is checked in step S2311. If Yes in step S2311, the next frame data is set as current frame data in step S2312, and the flow returns to processing of step S2302; if No in step S2311, a series of processes end.

If it is determined in step S2304 that processing did not end within the limit time, the flow shifts to step S2313, and only decoded data are written in the memory. The display timing complies with time management information in the packet header. Since decoding processing ends within the limit time, the display time defined in the header coincides with the actual data display time. This relationship will be explained with reference to Figs. 24A to 24D.

Fig. 25A shows 2x-speed playback, similar to

Fig. 18B. The first packet received at timing t = t00 takes the time D1 for decoding processing and is displayed for the time C1. The second packet received at timing t = t20 takes the time D2 for decoding processing, but all the data cannot be decoded within this time because D2 > (t40-t20).

Fig. 25A shows an entire encoded stream which should be originally decoded, and Fig. 25B shows data decoded partway owing to a short processing time. In this example, data can only be decoded up to the intermediate bit plane BitS-2 in the intermediate subband HH2. Therefore, the image of this frame becomes smaller in image size than other frames.

Fig. 24C is a view for explaining the image size in 2x-speed playback. The image size of the second frame is smaller than an image size in normal playback in Fig. 24B. In practice, the image size rarely sequentially changes within one scene. The image has a small image size in a scene having a large code amount, like Fig. 24D, and a standard image size in a scene having a small code amount. Even if the image size changes, the number of frames to be played back does not decrease, and a smooth motion can be reproduced similarly to normal playback in Fig. 24B. In this manner, if the processing time is short, only decoded data are written in the memory to guarantee the image quality of at least the decoded data without decimating frames.

Fig. 27A shows 4x-speed playback. As for the first packet received at timing t = t00, all the data cannot be decoded within the time D1, and only data which can be decoded till the next packet reception timing t = t10 are displayed for the time C1. This also applies to D2, D3, and D4. In this example, since all the four packets cannot be decoded within their processing times, the images of the frames are displayed with a small size, as shown in Fig. 27B. However, the number of frames to be played back does not decrease even at a 4x speed, so a smooth motion can be reproduced.

As described above, image continuity can be maintained by the above processing for each packet though the size changes between frames.

[Fifth Embodiment]

The fourth embodiment changes the image size and maintains the number of frames to be played back. The fifth embodiment limits the image size to be changed and can guarantee a predetermined image size.

A decoding apparatus in the fifth embodiment has the same arrangement as that in Fig. 1. Decoding processing in the fifth embodiment will be explained with reference to the flow chart of decoding processing of the decoding apparatus in the fifth embodiment shown in Fig. 26. Note that a description of the same processing as in the flow chart shown in Fig. 23 will be omitted. Assuming that processing of step S2301 has already ended in processing of step S2601, processing of step S2601 is not shown in Fig. 26. If the end of decoding of the current data is determined in step S2602, the same processing as processing from steps S2305 to S2311 is executed.

Whether the processing time has reached the limit time is checked in step S2603. If Yes in step S2603, whether the image size of decoded data is larger than the smallest image size set in advance is determined in step S2604. The smallest image size is set in advance such that all data must be decoded up to the subband LL or data must be decoded up to a size of LL + HL2 + LH2 + HH2. The set smallest image size is preferably set in accordance with the processing performance of the decoding unit, which will be described later. If Yes in step S2604, the flow advances to step S2605, and the decoding result is written in a memory 104a.

The following processing is the same as processing in step S2310 and subsequent steps in Fig. 23. That is, the flow waits for the reception timing of the next frame data in step S2606. If the next frame data exists in step S2607, the next frame data is set as current frame data in step S2608, and decoding processing continues. If No in step S2607, the current frame data is determined as a final frame, and a series of processes end.

If No in step S2604, the flow advances to step S2609 and waits for the reception timing of the next frame data. In step S2610, whether the next frame data exists is checked. If Yes in step S2610, the flow shifts to step S2611, and the limit time is changed. The limit time is changed by newly adding time management information within the packet header. That is, the current frame is processed within the total time of the processing time of the current frame and the processing time of the next frame. In step S2612, the next frame data is discarded, and data in a decoding buffer 102a is held.

This relationship will be explained by exemplifying 4x-speed playback in Figs. 28A and 28B. In decoding processing of the first frame in Fig. 28A, all encoded data cannot be decoded, but the image size is larger than the smallest image size, and the first frame image is displayed for the time C1. In decoding processing of the second frame, data cannot be decoded even with the smallest image size within the limit time D2, so the decoding time of the second frame is assigned up to the decoding limit time t of the third frame = t30. The display time is also prolonged to C2 + C3, which corresponds to two frames. In other words, the third frame image is not displayed, but the image size of the second frame image is held.

When the smallest image size is not guaranteed, the frame frequency (the number of frames displayed per second) increases as the speed of high-speed playback increases. In some cases, the image size is preferably maintained even by decimating frames in terms of the processing performance of the decoding unit, the display performance of the display, and visual effects. The data amount per frame and display frame rate have the relationship shown in Fig. 31. To increase the display frame rate, the data amount per frame must be decreased. To increase the data amount per frame, the display frame rate must be decreased. The data amount and image quality have a relationship as shown in Fig. 30. In this case, decreasing the data amount per frame decreases the image size. The smallest image size is set by balancing the smoothness of the frame motion and the image size of each frame. As a simple implementation means, the relationship between the playback speed of high-speed playback and the smallest image size is held as a table. Fig. 33 shows an example of the table. In this example, the full image size is guaranteed up to a 3x speed, and a 1/4 image size is guaranteed up to a 6x speed. For a speed higher than the 6x speed, a 1/8 image size is guaranteed.

As described above, the decoding apparatus of the fifth embodiment always manages the smallest image size of a frame to be decoded and always generates an image with a predetermined or higher image size.

[Sixth Embodiment]

The third embodiment prevents the image at the same tile position from successively failing in decoding. The sixth embodiment gives high priority to tiles around the center of the screen so as to obtain an image with a stable size around the center of the screen.

A decoding apparatus in the sixth embodiment has the same arrangement as that in Fig. 1. Decoding processing in the sixth embodiment will be explained with reference to the flow chart of Fig. 29. A description of the same processing as that in the flow chart shown in Fig. 2 will be omitted. In the case, the processing unit is not a frame but a tile.

The first band of the first tile in the current frame is processed for each plane (loop of steps S2903 → S2904). After processing of all the planes ends, processing is repeated for the next band (loop of steps S2905 → S2906). After processing of all the bands ends, processing is repeated for the next tile (loop of steps S2907 → S2908). This processing is repeated for all the tiles in the frame.

In processing in units of tiles, the image size of the tile may be localized owing to the difference in data amount between tiles on the screen in high-speed playback. This is prevented by giving priority to the playback position of a tile image and decoding tiles in descending order of priority. A tile near the center of the screen is given high priority, and a tile at the end of the screen is given low priority. Accordingly, a stable-size image can be attained near the center of the screen.

As described above, decoding processing of the decoding apparatus in the sixth embodiment enables decoding the image at the same position with a stable size by giving priority to the decoding tile position in playing back a decoded frame.

[Other Embodiment]

The following description concerns a case wherein the decoding apparatus shown in Fig. 1 according to the above embodiments is implemented by using a general-purpose personal computer.

Fig. 34 is a block diagram showing the hardware arrangement of a personal computer 50 functioning as the decoding apparatus shown in Fig. 1. As shown in Fig. 34, the personal computer 50 comprises the following arrangement in addition to a CPU 1 for controlling the overall system, a ROM 2 which stores a boot program and the like, and a RAM 3 functioning as a main memory.

An HDD 4 is a hard disk device. A VRAM 5 is a memory where image data to be displayed is mapped.

Image data or the like can be mapped in the VRAM 5 and displayed on a CRT 6. Reference numeral 7 denotes a keyboard and mouse for performing various settings.

As shown in Fig. 34, the HDD 4 stores an OS, the tables in Figs. 32 and/or 33, and a decoding program corresponding to a flow chart shown in at least one of Figs. 2, 5, 20, 23, 26, and 29. The decoding program also includes the functions of the DEMUX unit 101, decoding units 103a and 103b, and sync controllers 106a and 106b in Fig. 1.

The RAM 3 also functions as the decoding buffers 102a and 102b and the memories 104a and 104b. The function of the output units 105a and 105b is implemented by the VRAM 5 and CRT 6. Designation of the playback speed, setting of the lowest image quality in the second embodiment, and/or setting of the smallest image size in the fifth embodiment is input via the keyboard and mouse.

The OS and decoding program stored in the HDD 4 is loaded to the RAM 3 upon turning on the power supply, and executed by the CPU 1. The execution speed of decoding processing in this arrangement greatly depends on the processing performance of the CPU 1. It is, therefore, preferable to create a table (see Fig. 32) representing the relationship between the playback speed and the lowest image quality so as to set the lowest image quality in accordance with the processing performance of the CPU 1 in the second embodiment described above. Similarly, it is preferable to create a table (see Fig. 33) representing the relationship between the playback speed and the smallest image size so as to set the smallest image size in accordance with the processing performance of the CPU 1 in the fifth embodiment described above.

This arrangement allows the personal computer to function as the decoding apparatus according to the present invention and perform processing of the above-described embodiments.

The present invention may be applied to a system constituted by a plurality of devices (e.g., a host computer, interface device, reader, and printer) or an apparatus comprising a single device (e.g., a copying machine or facsimile apparatus).

The object of the present invention is also achieved when a storage medium (or recording medium) which stores software program codes for realizing the functions of the above-described embodiments is supplied to a system or apparatus, and the computer (or the CPU or MPU) of the system or apparatus reads out and executes the program codes stored in the storage medium. In this case, the program codes read out from the storage medium realize the functions of the above-described embodiments, and the storage medium which stores the program codes constitutes the present invention. The functions of the above-described embodiments are realized not only when the computer executes the readout program codes, but also when the operating system (OS) running on the computer performs part or all of actual processing on the basis of the instructions of the program codes.

The functions of the above-described embodiments are also realized when the program codes read out from the storage medium are written in the memory of a function expansion card inserted into the computer or the memory of a function expansion unit connected to the computer, and the CPU of the function expansion card or function expansion unit performs part or all of actual processing on the basis of the instructions of the program codes.

When the present invention is applied to the storage medium, the storage medium stores program codes corresponding to the above-mentioned flow chart (shown in at least one of Figs. 2, 5, 20, 23, 26, and 29).

As has been described above, the present invention can eliminate visual flaws between frames even if several frames cannot be decoded.

The present invention is not limited to the above embodiments and various changes and modifications can be made within the spirit and scope of the present invention. Therefore, to apprise the public of the scope of the present invention, the following claims are made.